TracNav

JPF Developer Guide

From the previous two sections, you have learned that JPF has one recurring, major theme: it is not a monolithic system, but rather a configured collection of components that implement various different functions like state space search strategies, report generation and much more. Being adaptive is JPF's answer to the scalability problem of software model checking.

This not only makes JPF a suitable system for research, but chances are that if are you serious enough about JPF application, you sooner or later end up writing extensions for it. This section should tell you how to do that, covering topics like:

- the JPF toplevel design

- basic mechanisms like ChoiceGenerators, partial order reduction and slot and field attributes
- the main extension points, namely Listeners, native peers (MJI) and bytecode factories
- common utility infrastructure like logging? and the report system
- how to invoke JPF from another program (e.g. an IDE)
- structure and directory layout of JPF projects and how to create them
- development guidelines like coding conventions and writing JPF tests